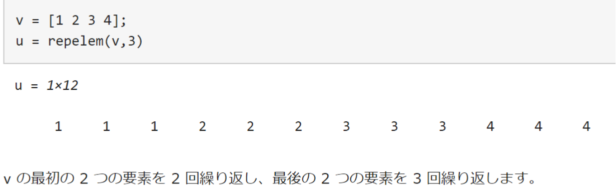
```
%SIMULATION CODE  - Weitzman, UrsuSeilerHonka 2022

function simulation = simWeitz(N_cons, N_prod, param,  seed)

%set seed for replication
rng('default'); rng(seed);
```



```
v = [1 2 3 4];
u = repelem(v,3)
```

```
u = 1×12

     1     1     1     2     2     2     3     3     3     4     4     4
```

v の最初の 2 つの要素を 2 回繰り返し、最後の 2 つの要素を 3 回繰り返します。

N_prod_cons=
[5,
5,
…,
5]

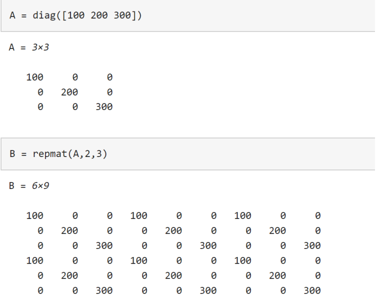
consumer =
[1,
2,
…,
1000,
1,
…,
1000]

```
%numb of observations
N_obs=N_cons*N_prod;
consumer=repelem(1:N_cons,N_prod); consumer=consumer';
N_prod_cons=repelem(N_prod,N_cons)';
```

prod=
[1,
2,
…,
5,
1,
…,
5,
]

```
%product id
prod=repmat([1:N_prod]',N_cons,1);

%outside option (represents option of not buying; is always searched) and brandFE
outside = (prod==1);
brand1=(prod==2);
brand2=(prod==3);
brand3=(prod==4);
brand4=(prod==5);
```



```
A = diag([100 200 300])
```

```
A = 3×3

   100     0     0
     0   200     0
     0     0   300
```

```
B = repmat(A,2,3)
```

```
B = 6×9

   100     0     0   100     0     0   100     0     0
     0   200     0     0   200     0     0   200     0
     0     0   300     0     0   300     0     0   300
   100     0     0   100     0     0   100     0     0
     0   200     0     0   200     0     0   200     0
     0     0   300     0     0   300     0     0   300
```

```
%prod char: in this case only brand intercepts
X=[brand1 brand2 brand3 brand4];
```

1000*1

```
[~,index_first,~]=unique(consumer,'stable');%first entry for each consumer e.g.[1, 6, 11]
index_last=cumsum(N_prod_cons);%last entry for each consumer e.g.[5, 10, 15]
```

```
%parameters
c=exp(param(end)).*ones(N_obs,1);%search cost
```

1000*1

```
xb=sum(X.*param(1:end-1),2);%utility from observables
```

1000*1

```
%draws affecting utility
epsilon=randn(N_obs,1);
```

1000*1

```
eta=randn(N_obs,1);
```

1000*1

```
%expected utility and utility
eut=(xb+eta).*(1-outside);
ut=eut+epsilon;
```

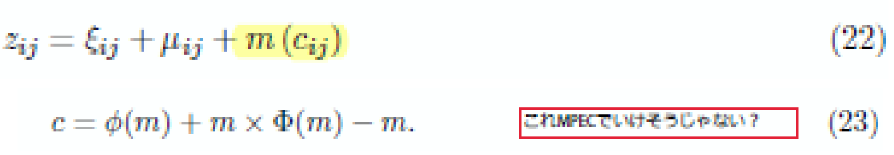
$$u_{ij} = \delta_{ij} + \varepsilon_{ij} = (\xi_{ij} + \mu_{ij}) + \varepsilon_{ij}, \qquad (4)$$
$$\varepsilon_{ij} \sim_{i.i.d} N(0, \sigma_\mu), \quad \mu_{ij} \sim_{i.i.d} N(0, \sigma_\varepsilon)$$

```
%%%%%%FORM Z's%%%%%%%%%
%%%1. look-up table method
table=importdata('tableZ.csv');
m=zeros(N_obs,1);
```

Zの計算は、lookup tableじゃなくて、DGPではちゃんと計算すべきでは？

```
for i=1:N_obs
    lookupvalue=abs(table(:,2)-c(i));
    if (table(1,2)>=c(i)&& c(i)>=table(end,2))
        [~,index_m]=min(lookupvalue);
        m(i)=table(index_m,1);
    elseif table(1,2)<c(i)
```

```
        m(i)=-c(i);
    elseif c(i)<table(end,2)
        m(i)=4.001;
    end
end
z=m+eut;
```



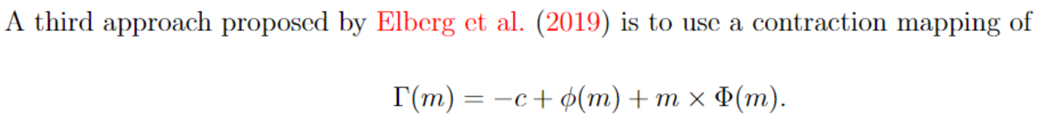
$$z_{ij} = \xi_{ij} + \mu_{ij} + m\left(c_{ij}\right) \tag{22}$$

$$c = \phi(m) + m \times \Phi(m) - m. \tag{23}$$

これMPECでいけそうじゃない？

```
% %%%2. newton method
% m=zeros(N_obs,1);
% x0 = 0; % initial point
% for i = 1:size(c, 1)
%     m(i) = newtonZ(c(i), x0);
% end
% z= eut + m;

% %%%3. contraction mapping method
% m=zeros(N_obs,1); % initial point
% for i = 1:size(c, 1)
%     m(i) = contractionZ(m(i), c(i));
% end
% z = eut + m;
```



A third approach proposed by Elberg et al. (2019) is to use a contraction mapping of

$$\Gamma(m) = -c + \phi(m) + m \times \Phi(m).$$

```
%plug in large value for outside option as it is always "searched" first
z=100000*outside+z.*(1-outside);


%order data by z
da=[consumer prod outside X eut ut z];
whatz=size(da,2);
whatu=whatz-1;
whateu=whatu-1;
```

da: 5000*10

列番号あてて る

consumer =
[1,
2,
...,
1000,
1,
...,
1000]

prod=
[1,
2,
...,
5,
1,
...,
5,
]

```
for i=1:N_cons
    [values(index_first(i):index_last(i)),
order(index_first(i):index_last(i))]=sort(da(index_first(i):index_last(i),whatz),'d
escend');
end
```

consumer iのzを高い順に並べてソート
valuesにsorted z
orderにsorted original index

1000*1

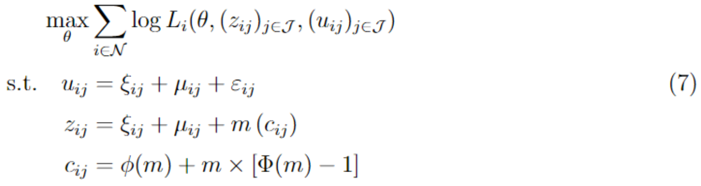
```
long_first = repelem(index_first',N_prod_cons); long_first=long_first(:);
order2=order'+long_first-1;

data=da(order2,:);
```

N_prod_cons=
[5,
5,
...,
5]

```
%search decision: 1. outside option always searched
searched=outside;

%search decision: 2. search if z greater than all ut searched so far (because z's
```



$$\max_{\theta} \sum_{i \in \mathcal{N}} \log L_i(\theta, (z_{ij})_{j \in \mathcal{J}}, (u_{ij})_{j \in \mathcal{J}})$$

$$\text{s.t.} \quad u_{ij} = \xi_{ij} + \mu_{ij} + \varepsilon_{ij} \tag{7}$$

$$z_{ij} = \xi_{ij} + \mu_{ij} + m\left(c_{ij}\right)$$

$$c_{ij} = \phi(m) + m \times [\Phi(m) - 1]$$

```
are ordered);
for i=1:N_cons
    %for every product, except outside option
    for j=index_first(i)+1:index_last(i)
        %max ut so far

relevant_ut_sofar=data(index_first(i):j-1,whatu).*searched(index_first(i):j-1,1);
        relevant_ut_sofar=relevant_ut_sofar(relevant_ut_sofar~=0);
        max_ut_sofar=max(relevant_ut_sofar);

        %search if z>ut_sofar
        if (data(j,whatz)>max_ut_sofar)
            searched(j)=1;
        end
    end
end

%transaction: among those searched, pick max ut
tran=zeros(N_obs,1);
searched_ut=data(:,whatu).*searched;
for i=1:N_cons
    A=searched_ut(index_first(i):index_last(i));
    A(A == 0)=-100000;
    [~,indexch]=max(A);
    tran(index_first(i)+indexch-1)=1;
end

%export data
length=repelem(N_prod,N_obs,1);%number of products per consumer
searched_mat=reshape(searched,N_prod,N_cons);
has_searched=searched_mat(2,:);%did consumer search at least once
has_searched=repmat(has_searched,N_prod,1);
has_searched=has_searched(:);
last=[zeros(1,N_prod-1) 1]';
last=repmat(last,N_cons,1);
                   =5
output=[data(:,1:whateu-1) last has_searched length searched tran];
save(sprintf('genWeitzDataS%d.mat',seed),'output');

end
```